IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:
Haugen, et al.

Serial No.: 09/813,255

Confirmation No.: 1709

Filed: March 20, 2001

For: Method and Apparatus for Refining an Alias Set of Address Taken Variables

§ Group Art Unit: 2122

§ Examiner: Kendall, C.

MAIL STOP AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

CERTIFICATE OF MAILING
37 CFR 1.8

I hereby certify that this correspondence is being deposited on January 13, 2005, with the U.S. Postal Service as First Class Mail in an envelope addressed to Mail Stop Amendments, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

January 13, 2005
Date

Gero G. McClellan

Dear Sir:

DECLARATION UNDER 37 C.F.R. § 1.131

We, the undersigned inventors, Patrick Todd Haugen and Tim Clayton Muehe, hereby declare as follows:

1. Attached is an inventor disclosure (Exhibit A) dated prior to January 9, 2001. Confidential information not relevant to the invention date of the present application is redacted.

2. In view of Exhibit A, the invention of pending claims 1-26 was conceived prior to January 9, 2001, and filed with due diligence from prior to January 9, 2001, to filing of the present application on March 20, 2001.

3. The undersigned Patrick Todd Haugen and Tim Clayton Muehe, hereby declare that all statements made herein of their own knowledge are true, and that all statements made upon information and belief are believed to be true. Further, that

statements made herein are made with the knowledge that willful false statements and the like are punishable by fine or imprisonment, or both, under 18 U.S.C. §1001, and that any willful false statements may jeopardize the enforceability of the above-entitled patent application, or any patents issued thereon.

1/11/2005
Date

Patrick Todd Haugen
Patrick Todd Haugen

1/11/2005
Date

Tim Clayton Muehe
Tim Clayton Muehe

[redacted]

Created By: Pat Haugen Created On: 07/12/2000 03:41:35 PM
Last Modified By: Pat Haugen Last Modified On: 09/19/2000 04:00:10 PM

[redacted]

[redacted]	Submitted
[redacted]	[redacted]
[redacted]	[redacted]
[redacted]	[redacted]
Submitted Date	09/19/2000 03:58:06 PM
[redacted]	[redacted]
[redacted]	[redacted]
[redacted]	[redacted]
[redacted]	[redacted]
[redacted]	[redacted]

[redacted]

Inventors: Pat Haugen/Rochester/IBM, Tim Muehe/Rochester/IBM

Inventor Name	[redacted]	[redacted]	[redacted]	[redacted]
[redacted] Haugen, Patrick T (Pat)				
Muehe, Tim C				

[redacted]

[redacted]

[redacted]

Main Idea

EXHIBIT A BEST AVAILABLE COPY

Title of disclosure (in English)

Alias refinement of arguments passed by reference to an inlined procedure

Idea of disclosure

1. Describe your invention, stating the problem solved (if appropriate), and indicating the advantages of using the invention.

This invention describes a method of improving the code generated by a compiler backend for arguments passed by reference to an inlined procedure. Passing procedure arguments by reference (i.e. passing the address of the argument to the called procedure instead of the value of the argument) in computer source code usually has the detrimental effect of inhibiting many compiler optimizations on the formal argument. Since the address of the argument is passed to the caller, many compilers will lump the argument into an "address taken" alias set. Inclusion in this pessimistic alias set limits the optimizations that can be performed by the compiler backend since actions such as indirect stores (store through a pointer) and procedure calls will usually kill all items in the "address taken" alias set. Small accessor functions in object oriented languages such as C++ are a prime candidate for this type of optimization since they are called frequently and most often implemented as inline functions

This invention involves tracking where the address of the argument flows and replacing the indirect references within the inline expansion with direct references to the formal argument. If all such references can be replaced, the formal argument can be removed from the "address taken" alias set, allowing the backend greater opportunities for optimizations such as redundant load elimination, store elimination, etc.

Following is a fragment of C code which demonstrates the problem.

```
int *intPtr;
int proc1();

inline void proc2(int *i) {
      i++;
}

int proc3 ()  {
      int a;
      a = proc1();
      proc2(&a);
      *intPtr = 1;
      return (a);
}
```

In this example proc2 has a single integer parameter which is passed by reference. Since 'a' is passed to proc2 its address is passed on the call and variable 'a' will be placed in the "address taken" alias set. The indirect store through 'intPtr' will kill everything in the "address taken" alias set which means 'a' will be written to storage before the store to 'intPtr' and must be reloaded from storage for the return statement.

2. How does the invention solve the problem or achieve an advantage,(a description of "the invention", including figures inline as appropriate)?

A compiler backend that can replace references to 'i' in the inline expansion with direct references to 'a' would be able to eliminate the need for taking the address of 'a' and therefore remove it from the "address taken" alias set. An intermediate representation of the above example may look something like the following coming into the compiler backend.

```
@S1  a = proc1();
CALL   proc1          // Call procedure 'proc1'
STR    a              // Store return value into 'a'
@S2  proc2(a);
```

```
LDA    a              // Load address of 'a'
STR    i              // Store address of 'a' into inline parameter 'i'
@S3  i++;
LOD    i              // Load 'i'
IND    int*           // Load integer value pointed to by 'i' (indirect load of 'a')
INC    1              // Increment value
LOD    i              // Load 'i'
STO    int*           // Store incremented integer value at location pointed to by 'i' (indirect store to 'a')
@S4  *intPtr = 1;
LOD    1              // Load value 1
LOD    intPtr         // Load 'intPtr'
STO    int*           // Store 1 at location pointed to by 'intPtr'
@S5  return (a);
LOD    a              // Load 'a'
RET                   // Return
```

Both 'a' and the shadow symbol 'int*' would be included in the "address taken" alias set which means any reference to 'int*' will be seen as a possible reference to 'a'. This alias relationship will cause the compiler backend to perform the following actions which are not really necessary.

1) 'a' will get mapped to storage
2) The value for 'a' will be written out to storage at statement S1 since the indirect of load of 'int*' could be referring to it
3) The address of 'a' will be computed in statement S2
4) The references to 'int*' in statement S3 will result in a load from storage and a write back out to storage
5) 'a' will be loaded from storage in statement S5 since the indirect store to 'int*' in statement S4 could have modified it

By using this invention, a compiler backend would maintain information about where the value of the address of 'a' flows (references to 'i' in this example). For cases where that address is being used as the base address for an indirect load or store, the intermediate representation is changed to contain a direct reference to 'a'. If all such references are replaced then the backend can eliminate the statement to load the address of the argument and store it in the inline parameter (statement S2), and can also remove the argument from the "address taken" alias set. This would leave a new intermediate representation which looks like the following.

```
@S1  a = proc1();
CALL   proc1          // Call procedure 'proc1'
STR    a              // Store return value into 'a'
@S3  i++;
LOD    a              // Load 'a'
INC    1              // Increment value
STR    a              // Store incremented value into 'a'
@S4  *intPtr = 1;
LOD    1              // Load value 1
LOD    intPtr         // Load 'intPtr'
STO    int*           // Store 1 at location pointed to by 'intPtr'
@S5  return (a);
LOD    a              // Load 'a'
RET                   // Return
```

Since 'a' and the shadow symbol 'int*' are no longer aliased, standard compiler optimizations will be performed with the net result being that 'a' will reside in a register for it's lifetime, eliminating 4 storage

references ([redacted]) that would have been present before applying this invention.

3. If the same advantage or problem has been identified by others (inside/outside IBM), how have those others solved it and does your solution differ and why is it better?
Procedure inliners that are integrated in the compiler frontend have the ability to perform formal argument replacement at the time of inlining and hence do not generate code to load the address of the argument or place the argument in the "address taken" alias set. But for inliners that run as a separate step after the compiler frontend or as part of the compiler backend, something must be done in the compiler backend to solve the problem. Implementing the solution in the compiler backend means that all compiler frontends targeting that backend will benefit.

[redacted]

FURTHER MATERIAL REDACTED

FURTHER MATERIAL REDACTED

FURTHER MATERIAL REDACTED

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:
Haugen, et al.

Serial No.: 09/813,255

Confirmation No.: 1709

Filed: March 20, 2001

For: Method and Apparatus for Refining an Alias Set of Address Taken Variables

§
§
§ Group Art Unit: 2122
§
§
§
§ Examiner: Kendall, C.
§
§
§
§
§

CERTIFICATE OF MAILING
37 CFR 1.8

I hereby certify that this correspondence is being deposited on January 13, 2005, with the U.S. Postal Service as First Class Mail in an envelope addressed to Mail Stop Amendments, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

January 13 , 2005
Date

Gero G. McClellan

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

STATEMENT OF COMMON OWNERSHIP

The present application (Serial No. 09/813,255; hereinafter the "Application") and Patent No. 6,173,444 were, at the time the invention of the Application was made, owned by the same entity, or subject to an obligation of assignment to the same entity.

Respectfully submitted,

Gero G. McClellan
Registration No. 44,227
MOSER, PATTERSON & SHERIDAN, L.L.P.
3040 Post Oak Blvd., Suite 1500
Houston, TX 77056
Telephone: (713) 623-4844
Facsimile: (713) 623-4846
Attorney for Applicant(s)